Proceedings of the
1st International Conference and Exhibition on Future RFID Technologies
Eszterhazy Karoly University of Applied Sciences and
Bay Zoltán Nonprofit Ltd. for Applied Research
Eger, Hungary, November 5–7, 2014. pp. 79–92

doi: 10.17048/FutureRFID.1.2014.79

Special requirements on ontology for customer support in Internet of Things

László Kovács[a], **Gábor Kusper**[b]

[a]University of Miskolc
kovacs@iit.uni-miskolc.hu

[b]Eszterhazy Karoly University of Applied Sciences
gkusper@aries.ektf.hu

Abstract

The Internet of Things provides a promising architecture for application of smart devices. The usage of smart devices should be supported by an intelligent customer support module. This paper presents a high level model of custom support module in Internet of Things architecture. This work presents detailed reviews on the related areas, including ontology models on event and time management. The perfomed analysis of the existing ontology models for event and time oriented systems showed that no one of them provides the required functionality of the target system.

1. Introduction

The current developments in IT segment enable the extension of computer networks to attach many new kinds of objects. The sensors and optional small processor units on the everyday objects can collect a huge amount of data and these objects optionally can be used as special output devices too. The smart objects connected to the internet constitute the new architecture of 'Internet of Things' (IoT). The objects have a unique identity and they can communicate with each other and with the environment. The intelligence of the things are usually stored in a central or distributed knowledge base. The corresponding knowledge base should provide more functionality than the standard databases, it should support among others efficient and high level information retrieval and decision making. The architecture of Internet of Smart Things can be considered as a catalyser of the development of

new Embedded Intelligence [1]

In this paper, the investigation focuses on a special functionality of the related business intelligence, namely the support of user's guide services. This service can be used as a special technical documentation. Technical documentation are of special kind of documentations and based on [2] its definition can be given as "'Technical documentation is a summarised term for all data that a user of a product can be provided with. Technical documentation aims in transferring knowledge to the future reader of the documentation."'. The main challenges in production of an efficient technical documentation are the followings: diversity of available data; a gap between Technical Documentation and Knowledge Management; the different understanding and treatment of Technical Documentation; Missing Reference Model. The main tasks of the intelligent User's Guide architecture are the following

- to help the users by providing appropriate information on the usage of the different smart devices and
- to control the usage of the devices.

The knowledge base of the intelligent IoT applications are usually stored in an ontology database. The ontology systems are used to perform explicit conceptualization of the problem domain [3]. An ontology framework describes the common accepted true facts in a standardized format. The main building atoms of an ontology are the concepts, the relationships among the concepts and the properties of the concepts. The concepts are organized into a graph describing the different kinds of relationships. The current ontology models provide not only static storage but also some knowledge processing operators, too. The ontology model of OWL [4] contains a reasoning engine for integrity checking and for rule implication.

For description of the domain specific knowledge on the usage of smart objects, the related knowledge model should contain some special features which are not part of the regular ontology databases. Considering the functionality of the User's Guide Ontology, the following special elements can be highlighted:

- event management
- time management
- ownership management
- workflow management

The regular ontology model does not provide any special tools to work these components, usually some other methods are used to describe these aspects of the problem domain. The most widely known tool for management of temporal events is the Petri-net. The Petri net structure consists of state nodes and transition

nodes between the state nodes. There are also tokens in the system for status signaling. The firing of transitions corresponds to the events of the problem domains. The net can model the sequencing of the different events and can be used to restrict the set of allowed state transitions. The temporal logic is another formal tool to describe the logic of dynamic systems. The paper provides a survey on the different event-oriented ontology models including among others a relative new approach, the Time-Event Ontology Model. The analysis shows the shortcomings of the existing approaches and sets the main requirements on a new, more functional event-ontology model. Another component of our motivation is coming from the field of digital ownership. Digital ownership means that if someone buys a product, then he or she must be able to get also the information which describes the product. This means that there are sensitive data and events which influence the life cycle of these data.

The main goal of our investigation is to extend the traditional logic language of the ontology, the Descriptive Logic language with some new functional elements to support the required ontology functionality. The DL language is a widely supported logic framework to describe the constraints and implication rules in the ontology databases.

2. Ontology Models

The main goal of current ontology frameworks is to provide an reliable automatized tool to perform consistency checking, decision making and reasoning in ad-hoc decision problems. The main components of an ontology frameworks are the following modules [5]:

- storage layer: XML
- knowledge repository: OWL
- reasoning: DL
- query: SPARQL

The classic approach in knowledge representation is the application of binary valued logic. The true and false values give an efficient representation of the truth values. The crisp binary truth value is also a fundamental part of membership grade representation in the traditional conceptual modeling. Considering the classic ontology models [21] , the relationship between concept a and an another concept or value (as instance concept) b is a strict relationship. The methodology of classic FCA (Formal Concept Analysis) uses the same approach [22]. The context of the FCA is based also on a strict relationship between objects and attributes. This context can be represented also with a binary values attribute-set: the attribute a at object o is 1, if a(o) is true, otherwise the attribute a has a value 0. Thus every concept c can be uniquely represented with an attribute set, the intention part of

the concept, where this set is a crisp set. The main benefit of FCA model is that it provides an efficient framework for concept generalization. The generalization g of a concept c has an intention set which is subset of the intention set for c. Thus only the existence of some common properties are the key factors in the concept generalization process. Considering, some others classic software engineering design models, like Entity Relationship or UML, although the attributes have a multi-valued domain, the existence of an attribute at an entity is a strict binary valued parameter.

Description Logics (DL) [23] are considered the most important knowledge representation formalism for ontology unifying and giving a logical basis to the well known traditions of frame-based systems, semantic networks and KL-ONE-like languages, object-oriented representations, semantic data models, and type systems. The main advantages of using DL are the automatic classification, logical inference and consistency checking. It provides the most general approach to knowledge representation. This representation form provides a precise inference framework for reasoning purposes. It can be proven that the framework representation and the first order logic formalism are equivalent. Based on the researches of Brachman and Levesque [24] the reasoning in the frame and network representations can be accomplished without the full power of the first order logic. The DL provides a formal language for defining concepts and individuals [4]. A concept is an intentional description of a class of individuals. The binary relationships between the concepts are called roles. In DL, the basic logic operators are beside the general disjunction, conjunction and complement operators the inclusion (subsumption), classification and recognition. Concept C1 subsumes concept C2 when every instance of C2 is also an instance of C1. The classification integrates the concept into a taxonomy. The recognition determines the set of most specific concepts which an individual instantiates. There are some additional operators used in the extension languages like the union, role value restriction, role existence restriction, role number restriction, role transitivity, role hierarchy and inverse role. The value restriction constraints the range of role relationship. The standard approach in DL is the open terminology assumption which presumes an incomplete terminology.

The main operators of DL can be seen in Figure 1.

The concept graph of the ontology can be constructed in the RDF or OWL languages. In RDF [6], there are four building blocks in the model:

- resource : identification of concepts
- property: an attribute of a resource
- literal: constant text

The RDF graph is built up from triplets which represent an information atom an it contains three components: subject, predicate and object. The sentence *'Peter is reading a book'* can be represented with the following triplet:

	Syntax	Semantics
Individuals:		
individual name	a	$a^{\mathcal{I}}$
Roles:		
atomic role	R	$R^{\mathcal{I}}$
inverse role	R^{-}	$\{\langle x,y\rangle \mid \langle y,x\rangle \in R^{\mathcal{I}}\}$
universal role	U	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
Concepts:		
atomic concept	A	$A^{\mathcal{I}}$
intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
union	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
complement	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
top concept	$\top$	$\Delta^{\mathcal{I}}$
bottom concept	$\bot$	$\emptyset$
existential restriction	$\exists R.C$	$\{x \mid$ some $R^{\mathcal{I}}$-successor of x is in $C^{\mathcal{I}}\}$
universal restriction	$\forall R.C$	$\{x \mid$ all $R^{\mathcal{I}}$-successors of x are in $C^{\mathcal{I}}\}$
at-least restriction	$\geqslant n\,R.C$	$\{x \mid$ at least n $R^{\mathcal{I}}$-successors of x are in $C^{\mathcal{I}}\}$
at-most restriction	$\leqslant n\,R.C$	$\{x \mid$ at most n $R^{\mathcal{I}}$-successors of x are in $C^{\mathcal{I}}\}$
local reflexivity	$\exists R.Self$	$\{x \mid \langle x,x\rangle \in R^{\mathcal{I}}\}$
nominal	$\{a\}$	$\{a^{\mathcal{I}}\}$

Figure 1: Main operators

```
<#Peter, #to_read, #book>
```

As it can be seen a triplet can carry only a small fragment of the scenario and a set of related triplets is used to describe a complex domain. With the application of abstract relationships, the RDF graph can be used to represent higher level concepts too. For example, the concept graph for the sentence, *'Peter knows that Zoli drinks vodka'* is shown in Figure 2.

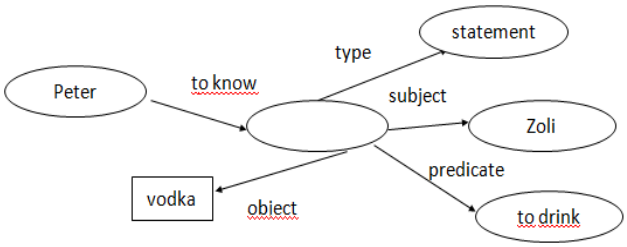

Figure 2: Main operators

As the RDF and RDF Schema standards does not support some key features of efficient modelling directly, an extension of RDF, the OWL ontology language is the dominant ontology language of current applications. The OWL language [7] includes among others the following new feature components:

- class declarations

- definition of class hierarchies
- extended assertions on classes
- class construction operators on classes
- support of enumerations
- domain and codomain definition for properties
- definition of property hierarchies
- constraints on properties
- support of individuals

The management of the OWL ontologies is supported by many many tools including reasoners like Pellet, Jena, RacerPro which provides a powerful API interfaces to connect them to application programs.

3. Event Ontology

The events play a core role in the management of activities and workflows. An event can characterised with some attributes, thus the event description should include several components. The set of involved components can vary in the different approaches, for example, the LODE [8] model defines only a base set of descriptors:

- event name
- timepoint of start
- duration
- location
- region
- involved agents
- involved objects

In the proposal of [10], a more formal approach is presented to establish an upper event-ontology in order-sorted logic. The sort hierarchy of the model corresponds to the hierarchy among the concepts. The model uses a dynamic event approach, where the dynamic behaviour is described with state change components. The constructed model contains a classification of event entities and event relationships. There are three main kinds of event relationships defined, the causal relationship covers the description for the cause and effect aspects, while next-event relationship is used to describe the temporal ordering of the events. The third kind of relationship is based on the spatial dependency of the actions. The proposed event type hierarchy is shown in Figure 3.

Event
NaturalEvent
Occurence1: ⟨*Time*, *Location*⟩
Occurence2: ⟨*Object*, *Time*, *Location*⟩
ArtificialEvent
Action1: ⟨*Agent*, *Object*, *Time*, *Location*⟩
Action2: ⟨*Agent*, *Time*, *Location*⟩
Action3: ⟨*AgentGroup*, *Time*, *Location*⟩
DynamicState
ObjectChange: ⟨*Object*, *Time*, *Location*⟩
EnvironmentChange: ⟨*Time*, *Location*⟩
StaticState
ObjectState: ⟨*Object*, *Time*, *Location*⟩
EnvironmentState: ⟨*Time*, *Location*⟩

Figure 3: Main operators

The paper presents a formal logic based formalism to describe the different relationships among the events. A special symbol is introduced for the state change:

$$F_1 \rightarrow OF_2$$

This symbol means that if F_1 is met and the event is executed then the expression F_2 will become true. Figure 4. shows an example how to describe an action that somebody drinks something:

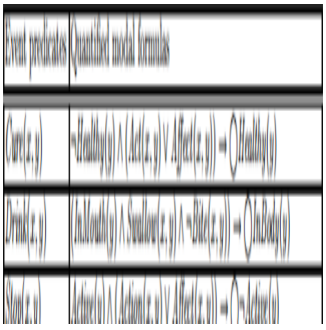

Figure 4: Example of event definition

Another event ontology model is given in [18] where an event is considered as a pattern of status change. The proposed description structure is given in Figure 5.

A key component is event modelling is the appropriate representation of the time concepts, i.e. the time points and the time durations. The roots of the time

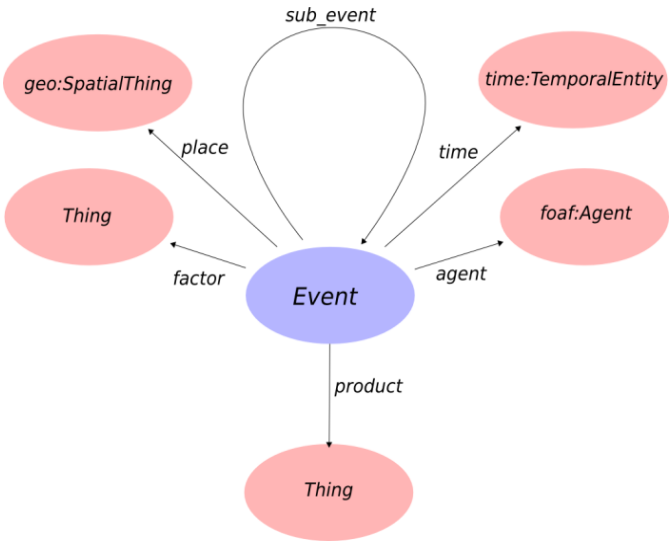

Figure 5: Example of event definition

ontology originate from the work of Allen [12]. Allen's model is based on the interval temporal logic and it defines 13 base temporal relationships among the intervals:

```
before, after, finishes, finished-by, overlaps, overlaped-by,
starts, started-by, during, contains, meets, met-by, equals
```

Later, this base model was extended on different ways to increase the precision and functionality of the description language. For example, in the paper [13], the 'before' relation was investigated in details and an extension of this operator was suggested to express the differences between the terms 'next before as cause' and the 'anytime before'. Another current approach of time ontology is presented in [14]. The main contribution of this approach is to argue for a zero length point of time representation instead of using finite length time intervals. The performed analysis shows that this approach can eliminate some important anomalies found in the Allen's model. In the paper [11], the extension of Allen's time ontology model is provided where the main new functionality refers to the definition of complex temporal expressions. For example, the term *"Every morning for the last four years"* is given with

```
(E T,t)[duration(T,*Year*) = 4 & ends(t1,T) &
iinside(t1,nowfn(D)) & everyp(S,{T},morning)]
```

Regarding the time ontology standards, the first project to provide temporal elements was the DAML ontology of time standard [15], published in 2002. This initial model was later extended to provide OWL functionalities. The working draft

version of OWL-Time [16] was published in 2006 and no final version is available. The model includes two basic concepts: instant and interval. The term instant means a point of time having zero length. The ontology contains two main element groups, the first relates to time points and time durations, while the second contains the base temporal relations similar to the Allen's model. An alternative standard is the OWL-Timeline model [17]. Every temporal concepts are related to a timeline concept which can be considered as coordinate system. The elements of OWL-Time are shown in Figure 6.

Property Name	Domain	Range
before	TemporalEntity	TemporalEntity
after	TemporalEntity	TemporalEntity
hasBeginning	TemporalEntity	Instant
hasEnd	TemporalEntity	Instant
inside	Interval	Instant
intervalEquals	ProperInterval	ProperInterval
intervalBefore	ProperInterval	ProperInterval
intervalMeets	ProperInterval	ProperInterval
intervalOverlaps	ProperInterval	ProperInterval
intervalStarts	ProperInterval	ProperInterval
intervalDuring	ProperInterval	ProperInterval
intervalFinishes	ProperInterval	ProperInterval
intervalAfter	ProperInterval	ProperInterval
intervalMetBy	ProperInterval	ProperInterval
intervalOverlappedBy	ProperInterval	ProperInterval
intervalStartedBy	ProperInterval	ProperInterval
intervalContains	ProperInterval	ProperInterval
intervalFinishedBy	ProperInterval	ProperInterval
years	DurationDescription	xsd;decimal
months	DurationDescription	xsd;decimal
weeks	DurationDescription	xsd;decimal
days	DurationDescription	xsd;decimal
hours	DurationDescription	xsd;decimal
minutes	DurationDescription	xsd;decimal
seconds	DurationDescription	xsd;decimal
hasDurationDescription	TemporalEntity	DurationDescription
unitType	DateTimeDescription	TemporalUnit
year	DateTimeDescription	xsd;gYear
month	DateTimeDescription	xsd;gMonth
week	DateTimeDescription	xsd;nonNegativeInteger
day	DateTimeDescription	xsd;gDay
dayOfWeek	DateTimeDescription	DayOfWeek
dayOfYear	DateTimeDescription	xsd;nonNegativeInteger
hour	DateTimeDescription	xsd;nonNegativeInteger
minute	DateTimeDescription	xsd;nonNegativeInteger
second	DateTimeDescription	xsd;decimal
timeZone	DateTimeDescription	tzont;TimeZone
inDateTime	Instant	DateTimeDescription
inXSDDateTime	Instant	xsd;dateTime
hasDateTimeDescription	DateTimeInterval	DateTimeDescription
xsdDateTime	DateTimeInterval	xsd;dateTime

Figure 6: Example of event definition

4. Requirements on Ontology Model of Customer Support

In analysis of a life cycle module for the Internet of Things, first the goals and operation of the engine should be investigated. The area of life cycle management of

IoT architecture is very new area in the IT world. In the literature only few articles and publications can be found addressing this problem domain. The whitepaper [19] summarizes the main technical challenges faced in this area. Among the main characteristics, two special features can be emphasised:

- data processing and operational decision making occurs near the object layer
- big-data analysis is performed at the server backend for strategic decision making

The main structure of the IoT life cycle engine is shown in Figure 7. [19].

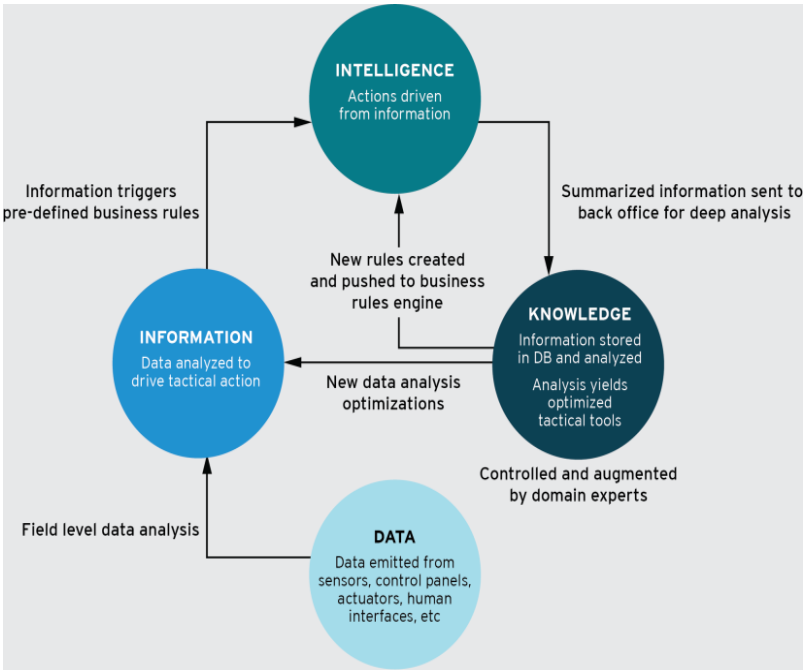

Figure 7: Example of event definition

Analysis in [20] focuses on the problem, how can the large traffic of data between the objects and central repositories reduced to provide an acceptable response time. In the study [21] on Internet of Things architecture a chapter is devoted to the life cycle management problem. The work categorizes the life cycle management tasks into the following domains:

- efficient scheduling

- service customization
- efficient data collection
- request optimization
- utility optimization
- service operation
- customer support

Our investigation focuses on the customer support module. This module requires a smart and very flexible behaviour as

- the user can be a human or a thing
- it should control the usage of the thing
- it should provide reasoning and decision making functionality
- it should provide user-tailored adaptivity
- flexible communication channel

Summarizing the ideas, the proposed customer support module should provide the following functionalities:

- identification of the thing and of the producer
- display the related history (linkage support)
- control of the usage (access rights, ownership, technical constraints)
- static and dynamic description of the usage context
- semantic queries
- support of different communication protocols
- smart decision making

Regarding the required knowledge repository on the server, a special ontology model should be implemented. Based on the workflow analysis presented in [9], the following aspects must be taken into consideration during the ontology description of a customer support engine:

- temporal properties
- event scheduling
- synchronization of events

- state change effect of the events
- exception handling
- roles of agents
- actor scheduling
- access control

Although this list contains the general base requirement elements, our special application area, namely the Internet of Things, involves additional elements into this requirement list. The following new elements can be considered as priority elements:

- event/action execution description
- object identification
- agent identification
- ownership control
- event history

Comparing the created list of requirements with the investigated ontology models found in the literature, we can see that no of them can cover all the required functionalities. The goal of the next phase in our research project is to work out the different ontology components based on the OWL ontology standard.

5. Conclusion

The paper presented a high level model of custom support module in Internet of Things architecture. The performed analysis of the existing ontology models for event and time oriented systems show that no one of them provides the required functionality of the target system. The proposed requirement and functionality list will be used in the next phase of the research project for detailed design of the prototype custom support system.

References

[1] Guo B.; Xi"an, Zhang D; Wang Z. *Living with Internet of Things: The Emergence of Embedded Intelligence*, Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing, 2011

[2] Sieber T. *Ontology-Based Modeling and Reuse of Technical Documentation*, PhD thesis, University of Miskolc, 2008

[3] Gruber T. *A Translation Approach to Portable Ontology Specifications*, Knowledge Acquisition, 5(2):199-220, 1993

[4] Bechhofer, S., F. van Harmelen, J. Hendler, I. Horrocks, D. McGuinness, P. Patel-Schneider, and L. Stein *OWL Web Ontology Language Reference, W3C Recommendation.*, 2004.

[5] Guarino, *Ontologies and ontological analysis: an introduction*, FOIS 2008, 2008

[6] Klyne, G. and J. Carroll, *Resource Description Framework (RDF):Concepts and Abstract Syntax*, W3C Recommendation., 2004

[7] Horrocks I, Patel-Schneider P, Van Harmelen F, *From SHIQ and RDF to OWL: The Making of a Web Ontology Language*, Journal of Web Semantics, 2003

[8] Shaw R, Troncy R, Hardman L, *LODE: An ontology for Linking Open Descriptions of Events* The Semantic Web, Lecture Notes in Computer Science Volume 5926, pp 153-167, 2009

[9] Barthelmess P, Wainer J., *Workflow Modeling* , COLLABORATION, COLLABORATIVE COMPUTING, Vol 1, pp. 61-86, 1994

[10] Kaneiwa K, Iwazume M, Fukuda K, *An upper ontology for event classifications and relations* AI'07 Proc. of Advances in artificial intelligence, pp. 394-403, 2007

[11] Hobbs, R. J. *A DAML Ontology of Time*, 2002

[12] Allen, J.F. , *Towards a general theory of action and time*, Artificial Intelligence 23, pp. 123-154., 1984

[13] Hemalatha M., Uma V., Aghila G, *Time ontology with Reference Event based Temporal Relations (RETR)*, International Journal of Web and Semantic Technology (IJWesT) Vol.3, No.1, 2012

[14] Schrag R, *Best-practice time point ontology for event calculus-based temporal reasoning*, Proceedings of STIDS, 2012

[15] Ferguson G, Allen J., *DAML-Time Homepage*, https://www.cs.rochester.edu/ ferguson/daml/, 2002

[16] Hobbs R., Pan F., *Time Ontology in OWL Homepage*, http://www.w3.org/TR/owl-time/, 2002

[17] Raimond Y., Abdallah S., *Timeline Ontology Homepage*, http://motools.sourceforge.net/timeline/timeline.html, 2007

[18] Raimond Y., Abdallah S., *Events Ontology Homepage*, http://motools.sourceforge.net/event/event.html, 2007

[19] Gigaom Research, *The information lifecycle for the Internet of Things*, https://gigaom.com/2014/10/28/the-information-life-cycle-for-the-internet-of-things/, 2014

[20] Ali, N.A., Abu-Elkheir, M. *Data management for the Internet of Things: Green directions*, Globecom Workshops (GC Wkshps), 2012 IEEE, 2012

[21] Uschold M, Gruninger M, *Ontologies: Principles, methods and applications* Knowledge Engineering Review, Vol 11, pp. 93-116, 1996

[22] Ganter ., Wille R., *Formal concept analysis:Mathematical Foundation*, Springer, 1999

[23] Benjamin N G, *Description Logic Programs: Combining Logic Programs with Description Logic*, ACM 2003

[24] Brachman R., Levesque H., *Readings in Knowledge Representation*, Morgan Kaufman, 1985